

Appl. No. 09/533,396  
Amdt. dated September 2, 2004  
Reply to Office Action of March 29, 2004

Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

Please amend claims 1 and 7 as follows:

1. (currently amended) System for providing traffic emulation for packet-switched networks, said system comprising:

an endpoint; and

an emulator module associated with said endpoint;

said emulator module comprising at least one finite state machine for modeling a number of traffic flows to be emulated, the traffic flows being emulated such that a constant number of processes, the number of processes being small relative to the number of traffic flows that can be emulated, can said emulator module operable to maintain simultaneous emulation of a relatively large the number of traffic flows using a constant number of processes, the number of traffic flows being larger than the constant number of processes, the number of processes being small relative to the number of traffic flows that can be emulated, the number of processes required for emulation being independent of the number of traffic flows to be emulated.

2. (original) The system according to Claim 1, wherein said emulator module comprises an emulation manager, said emulation manager comprising a finite state machine for maintaining the status of an emulation operation.

Appl. No. 09/533,396  
Amdt. dated September 2, 2004  
Reply to Office Action of March 29, 2004

3. (original) The system according to Claim 2, wherein said finite state machine of said emulation manager comprises three stages, said three stages comprising an initialization stage, and emulation stage and a result reporting stage.

4. (original) The system according to Claim 1, wherein:  
said endpoint further comprises a network interface module for facilitating the communication of flows to a network; and  
said at least one finite state machine comprises a plurality of finite state machines, wherein each finite state machine corresponds to a different one of said flows.

5. (original) The system according to Claim 4, wherein said network interface module comprises:

at least one port for receiving data;  
at least one port for receiving signals;  
a first background process for managing the receipt of data at said at least one data port;  
and  
a second background process for managing the receipt of signals at said at least one signal port.

6. (original) The system according to Claim 1, wherein said emulator module further comprises an event scheduler, said event scheduler comprising an event queue.

7. (currently amended) Method of providing traffic emulation for packet-switched networks, said method comprising:  
providing an endpoint; and

Appl. No. 09/533,396  
Amdt. dated September 2, 2004  
Reply to Office Action of March 29, 2004

providing an emulator module associated with said endpoint;

said step of providing an emulator module comprising providing at least one finite state machine for modeling a number of traffic flows to be emulated, the traffic flows being emulated such that a constant number of processes, the number of processes being small relative to the number of traffic flows that can be emulated, can said emulator module operable to maintain simultaneous emulation of a relatively large the number of traffic flows using a constant number of processes, the number of traffic flows being larger than the constant number of processes, the number of processes being small relative to the number of traffic flows that can be emulated, the number of processes required for emulation being independent of the number of traffic flows to be emulated; and

modeling traffic flows with said at least one finite state machine.

8. (original) The method according to Claim 7, wherein:

said step of providing an emulator module comprises providing an emulation manager;

said step of providing an emulation manager comprises providing a finite state machine for maintaining the status of said emulation; and

said method further comprising maintaining the status of said emulation with said finite state machine of said emulation manager.

9. (previously presented) The method according to Claim 8, wherein:

said step of providing a finite state machine for maintaining the status of said emulation comprises providing three stages, said three stages comprising an initialization stage, and emulation stage and a result reporting stage; and

Appl. No. 09/533,396  
Amdt. dated September 2, 2004  
Reply to Office Action of March 29, 2004

said method further comprises:  
  
performing each of said three stages in the following sequence: initialization stage,  
emulation stage and result reporting stage; and  
  
thereafter returning to said initialization stage.

10. (original) The method according to Claim 7, wherein:  
  
said step or providing an endpoint comprises providing a network interface module for  
facilitating the communication of flows to a network; and  
  
said step of providing at least one finite state machine comprises providing a plurality of  
finite state machines, wherein each said finite state machine corresponds to a different one of  
said flows.

11. (original) The method according to Claim 10, wherein said step of providing  
a network interface module comprises providing:  
  
at least one port for receiving data;  
  
at least one port for receiving signals;  
  
a first background process for managing the receipt of data at said at least one data port;  
  
and  
  
a second background process for managing the receipt of signals at said at least one signal  
port;  
  
said method further comprising:  
  
managing the receipt of data at said at least one data port with said first background  
process; and

Appl. No. 09/533,396  
Amdt. dated September 2, 2004  
Reply to Office Action of March 29, 2004

managing the receipt of signals at said at least one signal port with said second background process.

12. (original) The method according to Claim 7, wherein said step of providing an emulator module comprises providing an event scheduler, said event scheduler comprising an event queue.